615 Topic Modeling

1.Read the data set and take a general look.

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stringr)
movies <- read.csv("movie_plots_with_genres.csv")
```

2: Extract the Plot columns into Title and Text. Since we will meet some strange condition like “Graviton: The Ghost Particle”.

```
movies1 <- movies %>%
  mutate(
    Title = str_extract(Plot, "^[^:]+"),  # Extracts the title up to the first colon
    PlotText = str_replace(Plot, "^[^:]+: ", "")  # Keeps only the plot description
  )
head(movies1)
```

```
##   row                             Movie.Name   Genre
## 1  31                  Pioneers of the West  western
## 2  87                      The Infiltrators   action
## 3 146        "Graviton: The Ghost Particle"   sci-fi
## 4 197 Moses: Fallen. In the City of Angels.   action
## 5 314                       The Slave Trade  history
## 6 448                             The 303rd  history
##
## 1
## 2
## 3
## 4 Moses: Fallen. In the City of Angels.  :  A tale of a fallen angel who was sentenced to a human li
## 5
## 6
##                     Title
## 1 Pioneers of the West
## 2     The Infiltrators
## 3              "Graviton
## 4                  Moses
## 5      The Slave Trade
## 6            The 303rd
```

```
##
## 1
## 2
## 3
## 4 Fallen. In the City of Angels.  :  A tale of a fallen angel who was sentenced to a human life sent
## 5
## 6
```

3. We change our idea that we want to find all same text in the variables of Text and Movie.Name and drop them, since even though we extract a new column, they will be duplicate.

```
movies2 <- movies %>%
  rowwise() %>%
  mutate(Plot = str_remove_all(Plot, fixed(Movie.Name)))

head(movies2)
```

```
## # A tibble: 6 x 4
## # Rowwise:
##     row Movie.Name                              Genre   Plot
##   <dbl> <chr>                                   <chr>   <chr>
## 1    31 "Pioneers of the West "                 western " :  Caught by the Piu~
## 2    87 "The Infiltrators "                     action  " :  A tight team of t~
## 3   146 "\"Graviton: The Ghost Particle\" "     sci-fi  " :  Science is on the~
## 4   197 "Moses: Fallen. In the City of Angels. " action  " :  A tale of a falle~
## 5   314 "The Slave Trade "                      history " :  Beginning with th~
## 6   448 "The 303rd "                            history " :  Ret. Col. Louis \~
```

4.Then we will tokenize the column plot and remove the stop words

```
library(tidytext)
tidy_movies <- movies2 %>%
  unnest_tokens(word, Plot)
data("stop_words")
tidy_movies <- tidy_movies %>%
  anti_join(stop_words, by = "word")
head(tidy_movies)
```

```
## # A tibble: 6 x 4
## # Rowwise:
##     row Movie.Name              Genre   word
##   <dbl> <chr>                   <chr>   <chr>
## 1    31 "Pioneers of the West " western caught
## 2    31 "Pioneers of the West " western piutes
## 3    31 "Pioneers of the West " western pony
## 4    31 "Pioneers of the West " western express
## 5    31 "Pioneers of the West " western rider
## 6    31 "Pioneers of the West " western dick
```

```
movie_words <- tidy_movies %>%
  count(row, word, sort = TRUE)
```

5.Count word occurrences for each movie plot and convert this into a DTM

```
movie_dtm <- tidy_movies %>%
  count(row, word, sort = TRUE) %>%
  cast_dtm(row, word, n)
movie_dtm
```

```
## <<DocumentTermMatrix (documents: 1077, terms: 14095)>>
## Non-/sparse entries: 46254/15134061
## Sparsity           : 100%
## Maximal term length: 17
## Weighting          : term frequency (tf)
```

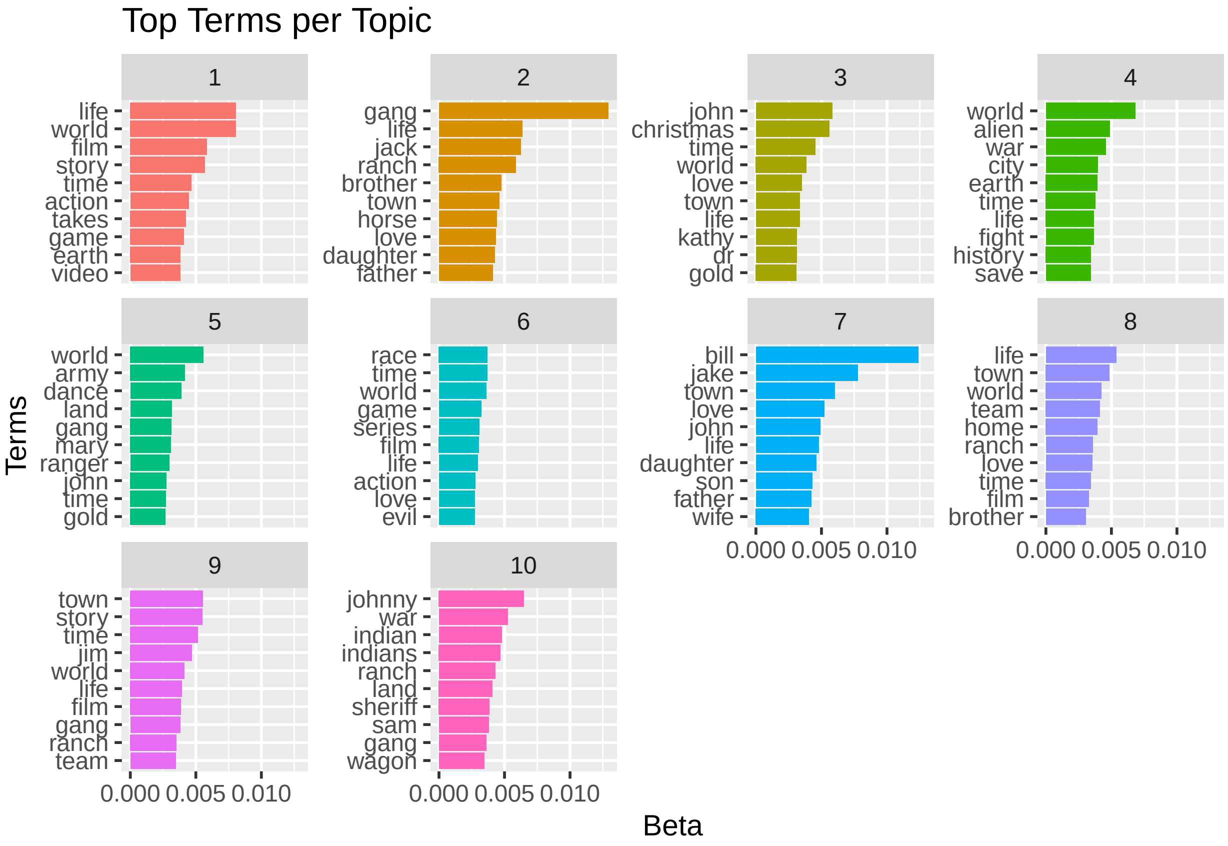
6. We will use a LDA function and fit the lda model.

```
library(topicmodels)
num_topics <- 10
lda_model <- LDA(movie_dtm, k = num_topics, control = list(seed = 1000))
```

7.Extract and visualize the top terms per topic.

```
top_terms <- tidy(lda_model, matrix = "beta") %>%
  group_by(topic) %>%
  slice_max(beta, n = 10) %>%
  ungroup() %>%
  arrange(topic, -beta)

ggplot(top_terms, aes(reorder_within(term, beta, topic), beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free_y") +
  coord_flip() +
  scale_x_reordered() +
  labs(title = "Top Terms per Topic", x = "Terms", y = "Beta")
```



8:Now we want to find the k means cluster and evaluate the cluster by find the distance between the centroid and each word.

```
dtm_matrix <- as.matrix(movie_dtm)

topic_distributions <- posterior(lda_model)$topics
k_clusters <- 5
set.seed(1000)
km <- kmeans(topic_distributions, centers = k_clusters)

movies$Cluster <- km$cluster

distances <- sapply(1:nrow(topic_distributions), function(i) {
  cluster_center <- km$centers[km$cluster[i], ]
  sqrt(sum((topic_distributions[i, ] - cluster_center)^2))
})

movies$Distance_to_Centroid <- distances

head(movies$Distance_to_Centroid)
```

```
## [1] 0.96037224 0.96031485 0.96037589 0.94475016 0.93846614 0.08442956
```

```
average_distances <- movies %>%
  group_by(Cluster) %>%
  summarize(Average_Distance = mean(Distance_to_Centroid))
print(average_distances)
```

```
## # A tibble: 5 x 2
##   Cluster Average_Distance
##     <int>            <dbl>
## 1       1           0.140
## 2       2           0.176
## 3       3           0.0817
## 4       4           0.848
## 5       5           0.148
```

9:Create a word cloud which the font size indicate the occurence of the words in each cluster.

```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
library(RColorBrewer)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.5
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

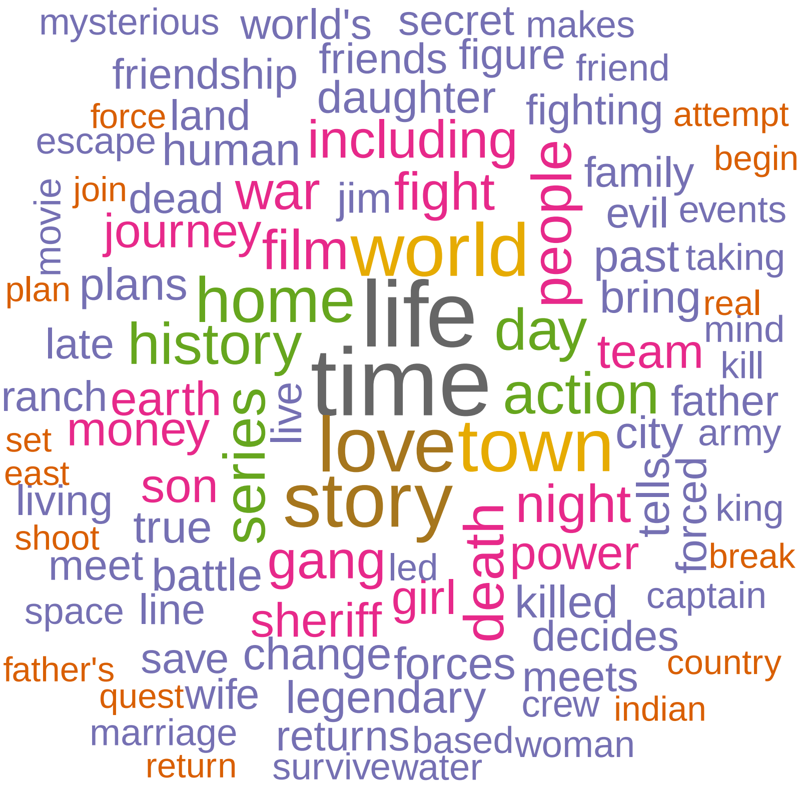
```
colors <- brewer.pal(8, "Dark2")

for (i in 1:k_clusters) {
  cluster_words <- movie_words %>%
```

```
    filter(row %in% movies$row[movies$Cluster == i]) %>%
    count(word, sort = TRUE)

  wordcloud(
  words = cluster_words$word,
  freq = cluster_words$n,
  min.freq = 2,
  max.words = 100,
  scale = c(3, 0.5),
  random.order = FALSE,
  colors = colors
)
}
```

```
## Warning in wordcloud(words = cluster_words$word, freq = cluster_words$n, :
## government could not be fit on page. It will not be plotted.
```



```
## Warning in wordcloud(words = cluster_words$word, freq = cluster_words$n, :
## dangerous could not be fit on page. It will not be plotted.
```

world
life
time
father
love
gang
war
death
town
money
takes
story
killed
meets
ranch
begins
discovers
friend
girl
local
team
brother
cattle
film
city
secret
sister
true
fight
dead
mysterious
life
world
death
war
love
story
home
girl
time
killed
battle
future
friend
gang
history
gang
world
time
life
story
takes
town
film
daughter
love
killed
ranch
family
father
friends
fight
day
war
set
world
time
life
town
story
war
film
local
girl
boss
journey
named
love
takes
ranch
forces
father
people
daughter
team